REMARKS

[0006] Applicant respectfully requests entry of the following remarks and reconsideration of the subject application. The remarks should be entered under 37 CFR. § 1.116 as they place the Application in better form for appeal, or for resolution on the merits.

[0007] Applicant respectfully requests reconsideration and allowance of all of the claims of the application. Claims 1-41 are presently pending. Claim 41 is amended herein. No claims are withdrawn or canceled herein. No new claims are added herein.

Formal Request for an Interview

[0008] If the Examiner's reply to this communication is anything other than allowance of all pending claims, then I formally request an interview with the Examiner. I encourage the Examiner to call me—the undersigned representative for the Applicant—so that we can discuss this matter so as to resolve any outstanding issues quickly and efficiently over the phone.

[0009] Please contact me to schedule a date and time for a telephone interview that is most convenient for both of us. While email works great for me, I welcome your call as well. My contact information may be found on the last page of this response.

Formal Matters

Claims

[0010] The Examiner objects to claims 19-23 for being unclear as to what the "means for" represents. Herein, Applicant submits that the means for is tied to hardware, including for example, an operating system 146(1) encapsulated by the framework 132 in Figure 2. Please see the corresponding discussion in the specification. Thus, Applicant respectfully requests that the Examiner withdraw the objections to these claims.

Substantive Matters

Claim Rejections under § 103

[0011] The Examiner rejects claims 1-41 under § 103. In light of the discussion during the above-mentioned Examiner interview, Applicant submits that these rejections are moot. For the reasons set forth below, the Examiner has not made a prima facie case showing that the rejected claims are obvious.

[0012] Accordingly, Applicant respectfully requests that the § 103 rejections be withdrawn and the case be passed along to issuance.

[0013] The Examiner's rejections are based upon the following references:

- **Chiang:** *Chiang,* US Patent Application Publication No. 2006/0294500 (Published December 28, 2006);
- **Roberts:** *Roberts, et al.,* US Patent No. 6,792,605 (issued September 14, 2004);
- **Firth:** *Firth, et al.,* US Patent No. 5,987,517 (issued November 16, 1999); and

- **Nevarez:** *Nevarez, et al.,* US Patent No. 6,609,158 (issued August 19, 2003).

Overview of the Application

[0014] The Application describes an application program interface (API) that provides a set of functions for application developers who build Web applications on Microsoft Corporation's .NET™ platform.

Cited References

Chiang

[0015] Chiang describes providing a web application and generating the basis for a complete web application source code. Based on user interface input files provided by graphic designers, the web application or generates application framework code, an event handler skeleton and s logic foundation code. Web developers then prepare additional source code object-oriented programming language based on the event handler skeleton and business logic foundation code to create web application business logic objects and handler methods. Ultimately the graphical user interface input files prepared by the designers and web application source code prepared by the web developers dynamically bound at runtime.

Roberts

[0016] Roberts describes accessing and using services and applications from a number of sources utilizing a customized application. The present invention accomplishes

this through an entity referred to as a web service. The web services architecture maintains a directory of services available to provide processing or services, along with the location of the services and the input/output schemas required by the services. When a request for data or services is received, appropriate services are invoked by a web services engine using service drivers associated with each service. A web services application is then generated from a runtime model and is invoked to satisfy the request, by communicating as necessary with services in proper I/O formats. In one embodiment, the web services application provides responses in the form of HTML that can be used to generate pages to a browser.

Firth

[0017] Firth describes a library of reentrant networking functions organized with file system semantics which allow a client application on a client computer connected to a computer network to establish communications with and exchange information with a server application on a server network computer. The library of reentrant networking functions are organized with file system semantics and parallel the function, structure and organization of a file system. Individual reentrant networking functions provide multiple networking features. The reentrant networking functions also provide asynchronous operations and security features. The library of reentrant networking functions can be included in, and called from multiple client applications. This library of reentrant networking function simplifies the creation of client applications such as network browsers that communicate with the Internet or an intranet computer network.

Nevarez

[0018] Nevarez describes connecting disparate software components in a network or other computer system. A language adapter maps programming language constructs to a general object-oriented interface. An object model adapter maps the general object-oriented interface to specific object models such as the Java/RMI model or a CORBA model. The system dynamically wraps non-object components, such as scripts, in an object wrapper as needed to make them accessible through the language adapter and the object model adapter. In this manner, a language-vendor-independent bridge is provided between components written in different programming languages, and the components also gain access to objects written in object models favored by various vendors.

Obviousness Rejections

Lack of *Prima Facie* Case of Obviousness (MPEP § 2142)

[0019] Applicant disagrees with the Examiner's obviousness rejections. Arguments presented herein point to various aspects of the record to demonstrate that all of the criteria set forth for making a prima facie case have not been met.

Based upon Chiang in view of Roberts

[0020] The Examiner rejects claims 1-8, 10-16, 19-22, 24-29, 31-34 and 36-39 under 35 U.S.C. § 103(a) as being unpatentable over Chiang in view of Roberts. Applicant respectfully traverses the rejection of these claims and asks the Examiner to withdraw the rejection of these claims.

Independent Claim 1

[0021] Applicant submits that combination of Chiang and Roberts does not teach or suggest at least the following features as recited in this claim (with emphasis added):

> "A software architecture implemented at least in part by a computing device for a distributed computing system comprising:
>
> a plurality of applications configured to handle requests submitted by remote devices over a network, wherein ***the plurality of applications are written in different programming languages***;
>
> ***an application program interface*** to present functions used by the plurality of applications to access network and computing resources of the distributed computing system; and
>
> ***a common language runtime layer that translates the plurality of applications written in different programming languages into an intermediate language***, the intermediate language being:
>
> ***executed natively by the common language runtime layer***; and
>
> configured to access resources or services requested by the remote devices, whereby a seamless integration between ***multi-language application development*** is allowed and a robust and ***secure execution environment*** for ***multiple programming languages*** is provided."

[0022] The Examiner indicates (Action, p. 3-4) the following with regard to this claim:

5. As to claim 1, Chiang teaches a software architecture implemented at least in part by a computing device for a distributed computing system comprising:

a plurality of applications configured to handle requests submitted by remote devices over a network, wherein the plurality of applications are written in different programming languages (Web Application 400 page 3 paragraphs 0033, "... input files..." page 3 paragraphs 0035/0036, Input 605 page 4 paragraph 0039);

an application program interface to present functions used by the plurality of applications to access network and computing resources of the distributed computing system (Application Framework 410 page 3 paragraph 0033); and

a common language runtime layer that translates the plurality of applications written in different programming languages into an intermediate language (Web Application Source Code Output 610), the intermediate language being: executed natively by the common language runtime layer ("The web application generator 205 generates the application framework code 605...regardless of the language format for the input files, because the input tags are interpreted in the same fashion for the different languages...") and configured to access resources or services requested by the remote devices whereby a seamless and robust integration between multi-language application development is allowed (Web Application Generator 205 page 4 paragraphs 0039 – 0044, page 5 paragraph 0050, page 6 paragraphs 0064, page 7 paragraphs 0068/0069).

Chiang is silent with reference to providing secure execution environment for multiple programming languages.

Roberts teaches providing secure execution environment for multiple programming languages ("...access control..." Col. 4 Ln. 36 – 38, Col. 6 Ln. 1 – 9, Ln. 47 – 63).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify the system of Chiang with the teaching of Roberts because the teaching of Roberts would improve the system of Chiang by providing the essential services of *identification and authentication* (*I&A*), *authorization*, and *accountability* where identification and authentication determine who can log on to a system, and the association of users with the software subjects that they are able to control as a result of logging in; authorization determines what a subject can do and accountability identifies what a subject (or all subjects associated with a user) did.

[0023] Applicant notes the combination of Chiang and Roberts was improperly used to reject claim 1 in a Final Office Action issued on October 5, 2007. Subsequent to the filed response on March 26, 2008, the Examiner removed the combination of Chiang and Roberts when addressing claim 1 in the Non-Final Office Action issued July 10, 2008. Therefore, it previously appears the Examiner was convinced that the combination of Chiang and Roberts does not properly address the features in claim 1. However, in this action, the Examiner has re-introduced the combination of Chiang and Roberts in the rejection of this claim. Again, similar to the arguments presented in the response filed on March 26, 2008, Applicant submits that the Examiner is incorrect in applying the combination of Chiang and Roberts.

[0024] In this Action, the Examiner equates "different programming languages" as recited in the claims, to input files that can be formatted in any "mark-up language" as

disclosed in Chiang Para [0038-0040]. The multiple mark-up languages as recited in Chiang are not equivalent to the "different programming languages" as recited in claim 1.

[0025] As seen in Fig 6 of Chiang and discussed in Para [0039-0040], a web application generator reads a set of web application screens as the input and generates web application source code as the output. The input files include tags corresponding to a mark-up language, and the same output code is generated regardless of the format of the input files, because the input tags are interpreted in the same fashion for the different languages (e.g. XML, HTML, etc.).

[0026] However, Chiang does not disclose "*a common language runtime layer that translates the plurality of applications written in different programming languages into an intermediate language*" as recited in claim 1.

[0027] In contrast, Chiang discloses that computer programmers use only a single programming language, such as the Java language, to write the underlying function of the web application. (Chiang Para [0026]) Furthermore, the web application source code output is generated as a corresponding Java class. (Chiang Para [0049]) This disclosure does not address the claimed element specifying "multi-language application development" and an "intermediate language being executed natively by the common runtime layer" as recited in claim 1.

[0028] Furthermore, the Roberts reference does not account for the deficiencies as explained with respect to Chiang.

[0029] As shown above, the combination of Chiang and Roberts does not teach or suggest all of the elements and features of this claim. Accordingly, Applicant asks the Examiner to withdraw the rejection of this claim.

Independent Claims 11 and 26

[0030] Independent claims 11 and 26 each include at least one feature similar to the claimed features as explained above with respect to claim 1. Thus independent claims 11 and 26 are allowable over the cited references for at least similar reasons as claim 1. Accordingly, Applicant asks the Examiner to withdraw the rejection of these claims.

Independent Claims 5, 19 and 31

[0031] In addition to the explanation given above regarding claim 1, with respect to these claims, Applicant submits that the combination of Chiang and Roberts does not teach at least the following elements as recited in these claims (with emphasis added):

- "***wherein the seamless integration allows for the ability to use a particular code module written in a first programming language with a code module written in a second programming language.***"

[0032] As shown above, the combination of Chiang and Roberts does not teach multi-language application development as recited in these claims. As a result, the cited references do not teach the "*ability to use a particular code module written in a first programming language with a code module written in a second programming language*"

as recited in independent claims 5, 19 and 31. Accordingly, Applicant asks the Examiner to withdraw the rejection of these claims.

Independent Claims 24 and 36

[0033] In addition to the explanation given above, Applicant submits that the combination of Chiang and Roberts also does not teach at least the following elements as recited in these claims as amended (with emphasis added):

- "…a common language runtime layer that allows seamless multi-language development, ***with cross language inheritance*** and translates the one or more software programs ***written in different programming languages into an intermediate language*** that is supported by the common language runtime layer..."

[0034] As shown above, the combination of Chiang and Roberts does not disclose "*different programming languages*" or an "*intermediate language*" as recited in these claims. As a result, the cited references do not disclose the "cross language inheritance" as recited in independent claims 24 and 36. Accordingly, Applicant asks the Examiner to withdraw the rejection of these claims.

Dependent Claims 2-4, 6-8, 10, 12-16, 20-22, 25, 27-29, 32-34 and 37-39

[0035] These claims each ultimately depend upon one of independent claims 1, 5, 11, 19, 24, 26, 31 or 36. As discussed above, claims 1, 5, 11, 19, 24, 26, 31 and 36 are

allowable over the cited references. It is axiomatic that any dependent claim which depends from an allowable base claim is also allowable over the cited references. Additionally, some or all of these claims may also be allowable for additional independent reasons.

Based upon Chiang in view of Roberts, and further in view of Firth

[0036] The Examiner rejects claims 9, 17-18, 23, 30, 35 and 40 under 35 U.S.C. § 103(a) as being unpatentable over Chiang in view of Roberts, and further in view of Firth. Applicant respectfully traverses the rejection of these claims at least because Firth does not account for the deficiencies in the combination of Chiang and Roberts as explained above with respect to claims 5, 11, 19, 26, 31 or 36, from which claims 9, 17, 23, 30, 35 and 40 respectively depend. Thus, Applicant asks the Examiner to withdraw the rejection of these claims.

[0037] Furthermore, Applicant submits that independent claim 18 includes at least one claimed feature similar to claim 1 explained above. As a result, independent claim 18 is allowable over the cited references for at least similar reasons as those explained above.

Based upon Roberts in view of Firth, and further in view of Nevarez

[0038] The Examiner rejects claim 41 under 35 U.S.C. § 103(a) as being unpatentable over Roberts in view of Firth, and further in view of Nevarez. Applicant

respectfully traverses the rejection of this claim and asks the Examiner to withdraw the rejection of this claim.

Independent Claim 41

[0039] Applicant submits that combination of Roberts, Firth and Nevarez does not teach or suggest at least the following features as recited in this claim (with emphasis added):

> "providing a common language runtime layer that ***translates Web applications written in different programming languages into an intermediate language,***
> ***the intermediate language being:***
> ***executed natively by the common language runtime layer***; and
> configured to access resources requested by the client applications;
> wherein, the different programming languages are selected from a plurality of programming languages, the plurality of programming languages comprising:
> Visual Basic; C++; C#; COBOL; Jscript; Perl; Eiffel; and Python."

[0040] The Examiner indicates (Action, p. 13-17) that the combination of Roberts, Firth and Nevarez teaches each feature as recited in this claim. Applicant respectfully disagrees.

[0041] The Examiner is correct when indicating (OA p. 15), that Roberts is silent to "…providing a common language runtime layer that translates Web applications written in different programming languages into an intermediate language, the

intermediate language being: executed natively by the common language runtime layer; and configured to access resources requested by the client applications; and wherein the different programming languages are selected from a plurality of programming languages, the plurality of programming languages comprising: Visual Basic; C++; C#; COBOL; Jscript; Perl; Eiffel; and Python."

[0042] The Examiner then relies upon Firth and Nevarez (OA p. 16-17) as accounting for part of the deficiencies in Roberts. Applicant respectfully disagrees. For example, Nevarez is directed towards connecting disparate software components in a network or other computer system. A language adapter maps programming language constructs to a general object-oriented interface. An object model adapter maps the general object-oriented interface to specific object models such as the Java/RMI model or a CORBA model. (Nevarez Abstract)

[0043] In Figure 2, Nevarez depicts a plurality of pure scripting languages (NSN 202, Perl 204, and Java Script 210) accessing objects under a universal component system (UCS) product 224 through a corresponding adapter, called an extension (indicated at 212, 214, 216, respectively). These extensions are examples of language template libraries. Each language template includes a specification that explains what a component in the language looks like to UCS, and the corresponding library implements that interface. (Nevarez Col. 9 Lines 40-49)

[0044] Thus, Nevarez teaches that for programming languages, separate adapters and/or templates are required in order to connect disparate software components. Nevarez does not teach or suggest "*providing* ***a common language runtime layer*** *that translates Web applications written in different programming languages into* ***an intermediate***

***language**, the intermediate language being: executed natively by the common language runtime layer"* as recited in claim 41.

[0045] As shown above, the combination of Roberts, Firth and Nevarez does not teach or suggest all of the elements and features of this claim. Accordingly, Applicant asks the Examiner to withdraw the rejection of this claim.

Dependent Claims

[0046] In addition to its own merits, each dependent claim is allowable for the same reasons that its base claim is allowable. Applicant requests that the Examiner withdraw the rejection of each dependent claim where its base claim is allowable.

Conclusion

[0047] All pending claims are in condition for allowance. Applicant respectfully requests reconsideration and prompt issuance of the application. If any issues remain that prevent issuance of this application, the **Examiner is urged to contact me before issuing a subsequent Action**. Please call or email me at your convenience.

Respectfully Submitted,

Lee & Hayes, PLLC
Representatives for Applicant

/Jacob Rohwer 61229/ Dated: 4/17/2009

Jacob P. Rohwer (jacob@leehayes.com; 206-876-6004)
Registration No. 61,229
Bea Koempel-Thomas (bea@leehayes.com; 509-944-4759)
Registration No. 58,213
Customer No. **22801**

Facsimile: (509) 323-8979
www.leehayes.com